OOPS :

Date / /
Page

OOPS is a (Programming model) & paradigm which revolves around Objects

Objects :- Real world instances of entities like class

Class = blueprint & template

Objects = instances of a class

Objects = instances

Class = car model

Object = A specific car of any model

Characteristics = Colours, chassis

Behaviours = How to start, How to change gear.

Concept of objects allows the oops model to easily access, use & modify the instance data & method

OOPS supported languages JAVA, C#, C++, Python ..etc

1) OOPS?

Programming paradigm that is defined using objects.

Objects = real world instances of entities.

2) Need of OOPS?

OOPS helps user to understand the software easily

→ readability, understandability, maintainability of

→ big software easily written & managed (or) easily using OOPS.

3) oops languages → Java, C++, Javascript, Python, PHP,

4) Other programming Paradigm other than oops?

1) imperative programming Paradigm
2) Declarative programming paradigm

1) imperative programming paradigm
It focuses on (How to execute program) (logic) and defines control flow as statements

1.a) Procedural programming paradigm :
→ read in order from top → bottom
→ Specifies the steps a program must take to reach the desired state.
eg: C, GLOBAL, FORTRAN, PLI, VB

1.b) OOP : organizes Programs as objects that contains some data & some behaviors

1.c) Parallel programming :
breaks a task into subtasks and focuses on executing them simultaneously at the same time.

2) Declarative programming paradigm :
focuses on (what to execute) and defines program logic.
(No control flows)

a) Logical programming paradigm : based on formal logic, refers to set of sentences expressing facts and rules about how to solve a problem.

b) Functional programming paradigm :
Programs are constructed by applying & composing functions.

Date / /
Page

c) Database programming paradigm :-
used to manage data and information structured as fields, records and files.
(Columns)

Programming paradigms

imperative Programming
- Procedural
- Object oriented
- parallel

Declarative programming
- logic Programming paradigm
- functional paradigm
- Database processing Approach

5. What is meant by structured programming? (top down)
programming which consists of a complete structured control flow
structures refers to flow, contains a set of rules
contains definitive control flow.
(C, C++, Java, Python)
All programming paradigm are structured programming paradigms.

6. Main features of oops?
→ Inheritance
→ Encapsulation
→ polymorphism
→ Data abstraction

7) advantages of oops:-
→ helpful in solving complex level of problems
→ easily handled, maintained
→ oops promote code reuse reduces redundancy
→ polymorphism offers lot of flexibility
→ (oops - bottom up approach)
→ hides unnecessary data

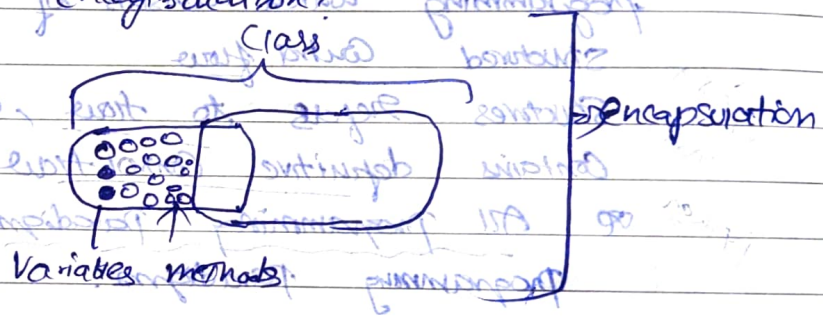
8) oops popular? better style of programming.

No Space

9) Class:
A template & blueprint contains some set of rules known as behaviours & functions one can create as many objects as they want.

10) What is an object?
Instance of the class.
Object consumes space & have some characteristic behaviours

11) What is encapsulation?



Method of putting everything that is required to do the job.
All the necessary data & methods are bind together. Unnecessarily details are hidden to the normal user.
Encapsulation can be defined in two different ways :-
1) Data Hiding :-
2) Data binding :- binding the data members and the methods together as a whole.

12) What is polymorphism?
Many forms
Something that have many jobs

Rhombus Triangle
Square
Shapes
Circle

Polymorphism refers to the process by which some code, data, method or object behaves differently under different circumstances

Polymorphism → Compile-time
→ Run-time

13)	Compile time polymorphism	Run time polymorphism
	Static (or) early binding	Dynamic or late binding
	→ happens at compile time	→ happens at runtime
	→ Compiler decides what shapes or value has to be taken	→ Shape or value depends on the execution
	→ Compile looks at the method signature & decides	

(15) Inheritance
Receiving some quality or behaviours from parent to an offspring, facilitates code reuse.

(16) What is Abstraction?
Wires of care need to be hidden so here abstraction helps.

(17) How much memory class occupy?
Class occupy no memory
Objects invoke classes and occupy memory

(18) Is it necessary to create objects from class?
No an object is necessary to be created if the base class has non-static methods

Date / /
Page

If The class has Static methods Then object don't need to be created

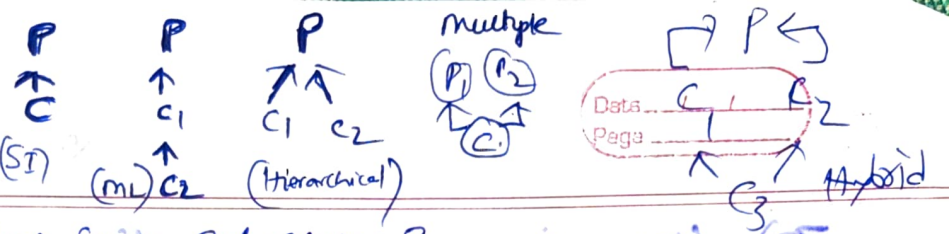
17) What is Constructor?
Serve as the special purpose of (Initializing The objects.)
helps to instantiate the member data and methods and assign them to objects

20 Constructors in C++ ? Default, parameterized Copy constructor,
destructor are free's up resources and memory occupied by an object.

23) diff B/w Class & Structure?
Structure is saved in The Stack memory
Class is saved in The heap memory
~~No~~ Data abstraction in Class

25. limitation of Inheritance?
* Inheritance needs more time to process
* Base Class, Child Class & highly coupled
* need for changes in nested Class also

26) Type of Inheritance?
1) Single Inheritance
2) Multiple Inheritance
3) Multi-level Inheritance
4) Hierarchical Inheritance
5) Hybrid Inheritance



27) What is Sub Class?

Subclass is a part of Inheritance it is an entity from another class.

28) Define Superclass?

part of inheritance, allows to inherit from itself

Class A (Base class)

↓

Class B (Derived class)

29) What is an Interface?

Special type of class Contains methods with no definition.

Cannot Create objects

30) Static Polymorphism

Compile time poly...

→ Compile time polymorphism can be achieved Through method overloading & operator overloading

31) Dynamic polymorphism

Run time poly...

Run time polymorphism can be achieved by method over-riding & functions decided during runtime.

32) Overloading vs overriding:

overloading: Compile time polymorphism feature has multiple implementation with same name eg: method, operator overloading

overriding: has the same name but different implementation during execution.

Date / /
Page

33) How is data abstraction accomplished?
Using abstract class & methods

34) Abstract Class :

abstact method :
non-abstact methods

36) Access Specifiers & Their Significance)
Private, Public, protected
places vital role in encapsulation.

37) What is an exception?
raised at runtime
brings execution to a halt
used for undesirable input.

38) What is exception handling?
Exceptions are the main reason for for
Software failure.
Try : Run Code
except : execute when there is an exception
else : No exception run this.
finally : Always run this Code

39) garbage Collection in oops lucets
it is for memory handling, Unwanted memory
is freed up by removing the objects.

40) Use can't run Java application without
oops concept
C++ can cause C-like Structural programming

C++, Python, Java, JS, Python
Pascal, Lua, Haskell

Data structures :

Date / /
Page

1) What is data structure ?
Way of organizing and storing data in a computer so that it can be accessed and manipulated efficiently.

2) Main types of Datastructure ?
arrays, linked list, stack, queues, trees, graphs and hash table.

3) what is an array ?
Collection of elements of same data type, stored in contigous memory location.

4) what is linked list ?
linked list is a data structure where each node contains a value and pointer to next element.

5) What is stack ?
A data structure following (LIFO) principle where elements are added and removed from the top.

6) what is queue ?
Follows (FIFO) principle, removed from the from added at the rear.

7) tree ? tree is a hierarchical datastructure composed of nodes where each node can have zero or more child nodes.

Date / /
Page

8. What is binary tree?
its a tree in which each node has at most two nodes referred to as left child & right child.

9. What is graph?
graph is collection of nodes and edges
edges represents relationship.

10. What is hash table?
uses hash function to map keys to values allowing for efficient retrieval & insertion.

11. Array VS linked list?
Array has fixed size, requires contigous memory, allow for random access,
linked list can dynamically grow and doesn't reequire contigous memory but won't support random access.

12. time complexity accessing an element in array = O(1)

13 Search in linked list = O(n)

14) What is doubly linked list?
Each node contains two pointers, one pointing to previous node & another pointing to next node.

15) What is circular linked list?
last node points back to the first node forming a loop.

Date / /
Page

16) Operations on Stack → Push()
Pop()
Peek() (or) top()

17) operations on Queue 1) enqueue (add)
2) dequeue (remove) 3) peek (view front)

18) What is BST?
BST is a Binary tree left child Contains value less than & equal to the node's value, right child Contains a value greater than the node's value.

19) What is an AVL Tree?
It's an a Self Balancing binary search tree in which height of left & right subtrees differ at most one.

20) What is hash function()?
takes input as key and produces hash code. Used to index the data in a hash table.

21) Collising handling in hash table?
two different keys generate same hash code techniques chaining (linked list) & open addressing

22) tc of Searching in a balanced BST? O(logn)

23) What is BFS?
BFS is a graph traversal explores all the vertices of a graph in breadth-first order, before moving to next level

Date / /
Page

24) What is DFS?
visits all descedents of a node before moving to the next sibling.

25) What is heap?
heap is a complete binary tree satisfies the heap property which means each parent node is either greater than or equal to (in a max heap) less than or equal to min heap.

26) Binary tree ⇒ atmost two children
BST ⇒ left child < root < right child

27) Time complexity of inserting an element into a heap?
O(log n)

28) removing root from heap = O(log n)

29) What is trie?
prefix tree, stores collections of strings
each node represents a common prefix.

30) Search in trie O(m) m = length(search.key)

31) what is dynamic programming?
technique of solving complex problems by breaking them down into simpler subproblems & solving them once storing the result to avoid redundant computations.

32) array vs matrix? Array is 1-D
whereas matrix is 2-D with rows & columns.

33) What is sparse matrix?
Matrix in which most elements are zero

34) What is Self-balancing tree?
It is a Binary Search tree that automatically adjusts its structures to maintain balance ensuring efficient search, insertion & deletion operations.

35) B-Tree?
Self-balanced tree that can have multiple keys & children per node, designed to reduce the number of disk access for large dataset.

36) TC Search in B-Tree O(log n)

37) Red-Black Tree?
It is a self-balancing binary search tree each node is coloured red or black & set of properties are maintained to ensure balance.

38) Search in Red-Black Tree? O(log n)

40) Search is hash table? TC = O(1)

42) Circular queue? last element points back to the first element, allowed for efficient memory utilization.

43) Priority queue is an abstract data type inserting elements with associated priorities and retrieving with highest/lowest priority

Date / /
Page

46) Stack = automatic storage of variables & functions call information
Heap is used for dynamic memory allocation

50) ~~47)~~ What is skip list?
Multiple parallel layers resembling linked list.

51) TC of searching in a skip list : O(log n)

52) What is disjoint set data structure?
A disjoint set data structure that keeps track of a collection of disjoint sets and supports efficient union and find operation.

54) What is suffix tree?
Represent all the ~~suffixes~~ suffixes of a given string in a compressed form.

56) What is a bit array?
Bit set or bit vector represents array of bits

~~58) What is bloom filter?~~

~~52~~) What is trie?
also known as prefix tree, stores collection of strings each node represents a common prefix

~~56~~) What is a monotonic stack?
elements either in non-increasing or non-decreasing order, allowing for efficient computations